

July 6, 2001

MULTI-LAYERED PACKET PROCESSING DEVICE**BACKGROUND OF THE INVENTION****1. Field of the Invention**

The present invention relates to a multi-layered packet processing device, and more particularly to a multi-layered packet processing device for processing
5 a received multi-layered packet at a high speed, by using multiple processors. The present application is based on Korean Patent Application No. 2000-56825, which is incorporated herein by reference.

2. Description of the Related Art

Based on a widely-known Open Systems Interconnection (OSI) reference
10 model, a precondition for data communication within a network is that the data has to be transferred from a top layer to a bottom layer and from the bottom layer to the top layer. During the data transmission from the top layer to the bottom layer, a header containing information is added to the data, through a process called 'encapsulation'.

15 In the communication field, the term 'encapsulation' means inclusion of a data structure in another data structure. Until reaching a target destination, the data structure is hidden. For example, when a transmission control protocol/internet protocol (TCP/IP) type data packet is encapsulated in an asynchronous transfer mode (ATM) frame, which is transferred on a cell basis,
20 the TCP/IP type data packet is only recognized as a bit stream among the ATM

data.

Functions of the devices in the network include processing a header/trailer of the encapsulated data, such as validation, conversion, updating, etc. Here, the processing needs to be handled fast for high-speed communication. Additionally, the trailer includes information about data length and Cyclic Redundancy Check (CRC).

FIG. 1 is a view for explaining a conventional packet processing. When the data packet is transferred to a memory (1-3) via an input interface (1-1), a Central Processing Unit (CPU; 1-4) reads, analyzes, and processes the header information stored in the memory (1-3), and outputs the packet via an output interface (1-2).

According to conventional data packet processing, since the processes of accessing and processing various information of the header of the packet stored in the memory (1-3), and storing the data in the memory (1-3) have to be repeatedly performed, the packet handling efficiency is deteriorated, while a next packet is held in stand-by state in the memory (1-3).

Accordingly, the conventional central packet processing method is inefficient as it relates to a packet processing speed.

SUMMARY OF THE INVENTION

The present invention has been made to overcome the above-mentioned problems of the related art, and accordingly, it is an object of the present invention to provide a multi-layered packet processing device capable of handling a received multi-layered packet at a hardware level and at a high speed, by using

multiple processors.

The above object is accomplished by a multi-layered packet processing device, according to the present invention, including an interface for transmitting and receiving a data packet with a node through a global network; and a plurality of packet processing portions for sequentially processing the data packet, in a pipeline pattern, according to a header of the data packet transferred through the interface.

BRIEF DESCRIPTION OF THE DRAWINGS

The above objects and other features and advantages of the present invention will become more apparent after a reading of the following detailed description in conjunction with the drawings, in which:

FIG. 1 is a view for explaining a conventional packet processing;

FIG. 2 is a block diagram showing a multi-layered packet processing device for an ATM transfer method, according to the present invention;

FIG. 3 is a block diagram showing a device added to the devices shown in FIG. 2;

FIG. 4A is a partial block diagram for explaining an operation of the IP processor of FIG. 3;

FIG. 4B is a flow chart for explaining the operation of the IP processor of FIG. 3;

FIG. 5A is a partial block diagram for explaining the operation of the GTP and UDP processors of FIG. 3;

FIG. 5B is a flow chart for explaining the operation of the GTP and the

UDP processors; and

FIG. 6 is a partial block diagram for explaining an operation of the IP processor of FIG. 3.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENT

5 The multi-layered packet processing device of the present invention will be described in greater detail below with reference to accompanying drawings.

The multi-layered packet processing device, according to the present invention, receives from a main module, via an interface, various types of packets which are encapsulated in ATM Adaptation Layer 5 (AAL5) protocol cell
10 formats. The Table 1 below lists the types and formats of the packets.

Table 1

General Classification	Description	Format
UMTS PLMN Core Network GGSN_incoming	Normal IP	IP(20) Payload
	Mobile IP (IP in IP)	MIP(20) IP(20) Payload
	Mobile IP (M in IP)	Modified IP(20) Min Header (8 or 12) Payload
	Mobile IP (Generic Routing Encapsulation(GRE))	
	Special UDP (Layer 2 Tunneling Protocol (L2TP), RADIUS, MIP registration, etc.)	IP UDP L2TP, etc. Payload
UMTS PLMN Core Network GGSN_outgoing or SGSN	Normal IP	IP (20) UDP (ignored) GTP(ignored) Payload
	GTP_c	IP (20) UDP (8) GTP (12) Payload
	GTP_u	IP (20) UDP (8) GTP (12) Payload
	Point to point protocol (PPP) (GTP with pre-defined TEID value)	IP (20) UDP (8) GTP (12) PPP Payload

The above Table 1 specifically lists the types and formats of the packets applied to a Universal Mobile Telecommunication System (UMTS).

5 The UMTS enables multimedia data transmission and reception through mobile devices having IP addresses, and employs a Home Agent (HA) for managing the mobile devices in a corresponding range, and a Foreign Agent (FA) for readjusting a data reception route according to the relocation of the mobile devices.

When the packet is transferred while being encapsulated in the ATM AAL5, as shown in the Table 1, the structure shown in FIG. 2 handles the packet
10 at a hardware level. FIG. 2 shows the structure of a main module interface 10, an ATM reassembly 20, an ingress IP processor 22, a GTP (GPRS Tunnel Protocol) and UDP (User Datagram Protocol) processor 40, a lookup processor 50, an ingress IP processor 24, a segmentation 60, and an ATM switch interface 70, all of which are connected in a pipelined structure.

15 The ATM reassembly 20 reassembles the ATM cells input through the main module interface 10 into a packet, and adds a tag to an area of the reassembled packet which is prepared for an attachment of the header, and outputs the reassembled packet as a first packet (P1).

The ingress IP processor 22 analyzes an IP header of the first packet (P1)
20 which is output from the ATM reassembly 20. When the destination address matches the system address, the ingress IP processor 22 excludes the IP header and outputs the packet as a second packet (P2) to the GTP and UDP processor 40. When the destination address and the system address do not match each other,

the ingress IP processor 22 outputs the first packet (P1) together with a bypass signal.

Upon receipt of the first packet (P1) with the bypass signal, the GTP and UDP processor 40 transfers the first packet (P1) to the lookup processor 50 without processing. Meanwhile, upon receipt of the second packet (P2), the GTP and UDP processor 40 checks a port number of the UDP header. If the second packet (P2) is determined to be a GTP packet as a result of checking the GTP header message type and Tunnel End Point ID (TEID), the GTP and UDP processor 40 looks up a valid index value and writes the valid index value in the tag, and outputs the packet as a third packet (P3) together with a lookup bypass signal. In this situation, a UDP header is not transferred.

Upon receipt of the third packet (P3) together with the bypass signal, the lookup processor 50 bypasses the signal without processing. Meanwhile, if the first packet (P1) is received, since it means the destination address of the received packet does not match the system address, the lookup processor 50 routes the destination address, and outputs a first packet (P1) updated with a destination address.

Upon receipt of either the first packet (P1) updated with the destination address, or the third packet (P3) having the valid index value written in its tag, the ingress IP processor 24 requests a remainder of a payload which is temporarily stored in the ATM reassembly 20, and outputs the remainder of the payload to the segmentation 60, after either the first packet (P1) updated with the destination address is output, or after the third packet (P3) having the valid index value

written in its tag is output.

When the remainder of the payload is received from the ingress IP processor 24, together with the first packet (P1) updated with the destination address, the segmentation 60 looks up an appropriate VPI/VCI value, segments the first packet (P1) into the ATM cells, and outputs the ATM cells to the main module interface 10. When the remainder of the payload is received together with the third packet (P3) having the valid index value written in its tag, the segmentation 60 segments the third packet (P3) into the ATM cells, and outputs the ATM cells with the tag to the ATM switch interface 70.

Here, the ATM reassembly 20 has two memory blocks, i.e., a connection memory 20-2 and a cell memory 20-1, as shown in FIG. 3. The cell memory 20-1 stores a remainder of the packet that is separated from the ATM reassembly 20. The GTP and UDP processor 40 has a TEID lookup table 40-1. The lookup processor 50 has an IP address lookup table 50-1. The segmentation 60 has an index table 60-1. The whole structure of the packet processing device is shown in FIG. 3.

The operation of the IP processor will be described with reference to FIGS. 3, 4A, and 4B.

Among the ATM cells transferred from the ATM reassembly 20 and through the main module interface 10, the first four (4) ATM cells, from which the headers are removed, are added to each other as data of 192 bytes. Then, added with the tag, the data of 192 bytes is transferred to the IP processor 30 as the first packet (P1), and accordingly, the IP processor 30 receives the first packet

(P1) (step S1). The IP processor 30 checks the IP header, and checks whether the IP destination address matches the address of the IP processor 30 (step S2). Meanwhile, the first packet (P1) is structured whereby the tag is at the beginning, followed by an IP 80, a UDP 8, a GTP 12, and the first part of the payload. When the addresses do not match each other, since the UDP 8 and the GTP 12 do not need to be processed, the IP processor 30 transfers the GTP_bypass signal to the GTP and UDP processor 40, together with the first packet (P1) (step S2-2). In this situation, the GTP and UDP processor 40 bypasses the first packet (P1) to the lookup processor 50 without processing. Meanwhile, when it is determined by the IP processor 30 that the addresses match each other, an IP header-removed second packet (P2) is transferred to the GTP and UDP processor 40 (step S3).

If the mobile IP packet is transferred, the mobile IP processor 22-1 analyzes the encapsulation type of the packet, and performs a decapsulation, according to the analyzed result. After decapsulating an outer IP header and checking on the encapsulated IP header, the packet is transferred to the next processor, i.e., the GTP and UDP processor 40 (step S3). Here, since the GTP and UDP processor 40 does not need to analyze the mobile IP packet, the packet is transferred together with the bypass signal.

Referring to FIGS. 5A and 5B, upon receipt of the second packet (P2) from the IP processor 30 (step S11), the GTP and UDP processor 40 checks if the packet is the UDP packet (step S12). If the packet is a GGSN (Gateway GSN)_outgoing and if the destination address matches the system address, the packet always has the IP/UDP/GTP format. Accordingly, if the packet is not a

which outputs a result only on an exact match occasion. Here, the exact match is required since mobile nodes within the UMTS network do not have topology-based prefixes.

Referring back to the FIGS. 4B and 6, the operation of the IP processor 30 will be described. When the ingress IP processor 24 receives the third packet (P3) from the lookup processor 50 (step S4), the ingress IP processor 24 checks whether the valid index value of the tag is set or not. When the valid index value of the tag is not set, the ingress IP processor 24 requests from the ATM reassembly 20, the remainder of the third packet (P3), by using the packet address field (step S5). When it is determined that the remainder of the third packet (P3) has been received (step S6), the third packet (P3) is transferred to the segmentation 60, and the remainder of the third packet (P3), which is requested by the ingress IP processor 24, is successively output to the segmentation 60, upon arrival to the ingress IP processor 24 (step S7). The other Start Of Packets (SOP) are not transferred to the segmentation 60 until the remainder of the third packet (P3) is completely transferred to the segmentation 60.

When the segmentation 60 receives the third packet (P3) having a valid index value written in its tag together with the remainder of the payload, the segmentation 60 outputs the tag-included ATM cells to the ATM switch interface 70. When the segmentation 60 receives the first packet (P1) updated with the destination address and the remainder of the payload, the segmentation 60 looks up an appropriate VPI/VCI value, segments the first packet (P1) into the ATM cells, and outputs the ATM cells to the interface 10.

Further, the packet processing device, according to the present invention, recovers the original IP header if the received packet was an encapsulated mobile IP packet. The packet processing device updates the IP header if the system was the packet's intermediate node to the final destination.

5 As described above, in the pipelined structure of the multiple processors for performing various functions, since the packet is transferred from one processor to the next processor without a processed header, the packet handling can be performed more efficiently. Further, by dividing the function module for sequentially processing the packet, more packets can be processed
10 simultaneously, and accordingly, data processing efficiency can be improved.

Accordingly, by using multiple processors in a pipelined structure for handling a received multi-layered packet, the packet can be processed fast at a hardware level.

As stated above, a preferred embodiment of the present invention is
15 shown and described. Although the preferred embodiment of the present invention has been described, it is understood that the present invention should not be limited to this preferred embodiment, but various changes and modifications can be made by one skilled in the art within the spirit and scope of the present invention, as hereinafter claimed.